L Number	Hits	Search Text	DB	Time stamp
1	15	"6374286"	USPAT; US-PGPUB; EPO; DERWENT; IBM_TDB	2004/02/03 10:43
2	566	(return$3 or releas$3 or unlock$3) same synchroniz$9 same thread	USPAT; US-PGPUB; EPO; DERWENT; IBM_TDB	2004/02/03 10:44
3	59	(return$3 or releas$3 or unlock$3) same synchroniz$9 same thread same pointer	USPAT; US-PGPUB; EPO; DERWENT; IBM_TDB	2004/02/03 10:54
4	2	("6411983").PN.	USPAT; US-PGPUB; EPO; DERWENT; IBM_TDB	2004/02/03 10:54

CiteSeer Find: Documents Citations

Searching for **PHRASE optimistic synchronization**.
Restrict to: Header Title Order by: Citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CS DBLP
110 documents found. **Order: citations weighted by year.**

Performance Evaluation of a Transactional DSM System - Wende, Schoettner.. (2002) (Correct) (7 citations)
DSM. Memory consistency is maintained by **optimistic synchronization** mechanisms and atomic transactions
past. Such a transaction based DSM with **optimistic synchronization** guarantees a sequential consistent view
www.plurix.de/publications/2002/pdpta2002.pdf

Threads and Input/Output in the Synthesis Kernel - Massalin, Pu (1995) (Correct) (59 citations)
synthesis, fine-grain scheduling, and **optimistic synchronization**. Kernel code synthesis reduces the
path for frequently used kernel calls. **Optimistic synchronization** increases concurrency within the
guir.cs.berkeley.edu/projects/osprelims/papers/synthesis.ps.gz

Parallel Simulation Today - Nicol, Fujimoto (1994) (Correct) (34 citations)
and dynamic memory management for **optimistic synchronization**. 1 Introduction Parallel simulation is
organizations, this machine is based on **optimistic synchronization**. The machine is a shared memory
www.cc.gatech.edu/computing/pads/PAPERS/parallel_sim_today.ps

Synthesis: An Efficient Implementation of Fundamental Operating.. - Massalin (1992) (Correct) (48 citations)
real-time data streams. ffl Lock-free **optimistic synchronization** is shown to be a practical, efficient
ftp.cs.columbia.edu/reports/reports-1992/cucs-039-92.ps.gz

Enabling Large-scale Simulations: Selective Abstraction.. - Huang, Estrin, Heidemann (1998) (Correct) (13 citations)
techniques such as conservative and **optimistic synchronization** mechanisms to maintain the correct
www.cs.utah.edu/~kwright/paper_summs/network_papers/../../papers/ns-abstractions.ps

A Lock-Free Multiprocessor OS Kernel - Massalin, Pu (1991) (Correct) (47 citations)
synchronization, developed from the **optimistic synchronization** methods developed for the
enter the critical section. Note that in **optimistic synchronization** the critical section should save enough
www.cs.columbia.edu/~library/TR-repository/reports/reports-1991/cucs-005-91.ps.gz

Region-Based Memory Management for Real-Time Java - Beebee, Jr. (2001) (Correct) (2 citations)
potential problem by using non-blocking, **optimistic synchronization** primitives throughout the
non-blocking synchronization primitives, **optimistic synchronization**, status variables, and atomic
www.flex-compiler.lcs.mit.edu/Harpoon/papers/wes-thesis.ps

A Scalable Mark-Sweep Garbage Collector On Large-Scale.. - Endo (1998) (Correct) (8 citations)
acquisitions are eliminated by using **optimistic synchronization**. With all these careful implementation,
this "lock-and-test" operation with **optimistic synchronization**. We tests a mark bit first and quit if
ftp.yl.is.s.u-tokyo.ac.jp/pub/papers/endo-mthesis-a4.ps.gz

Speculative Synchronization: Applying Thread-Level.. - Martínez, Torrellas (2002) (Correct) (1 citation)
sets our proposal apart from lock-free **optimistic synchronization** schemes of similar hardware simplicity,
our approach to two relevant lock-free **optimistic synchronization** schemes, and proposes Adaptive
www.csl.cornell.edu/~martinez/doc/asplos02.ps

IDES: A Java-based Distributed Simulation Engine - Nicol, Johnson, Yoshimura.. (1998) (Correct) (6 citations)
towards an optimistic method. A bevy of **optimistic synchronization** protocols exist, all complex and all
www.cs.dartmouth.edu/~nicol/papers/ides/mascots-ides.ps

NPSI Adaptive Synchronization Algorithms for PDES - Srinivasan, Reynolds, Jr. (1995) (Correct) (8 citations)
neither purely conservative nor purely **optimistic synchronization** algorithms will perform well
ftp.cs.virginia.edu/pub/techreports/CS-94-44.ps.Z

Time Management in the DoD High Level Architecture - Fujimoto, Weatherly (1996) (Correct) (8 citations)
platforms using conservative or **optimistic synchronizati n**. 5. federates using a mixture of event
www.cc.gatech.edu/computing/pads/PAPERS/HLA-PADS96.ps

On the Trade-off between Time and Space in Optimistic.. - Preiss, MacIntyre.. (1992) (Correct) (13 citations)
of communicating sequential processes. **Optimistic synchronizati n** means that the processes execute under
p r llel discrete event simul ti n. **Optimistic synchr niz ti n** ssumes th t the c ncurrent executi n

www.pads.uwaterloo.ca/Bruno.Preiss/pap[illegible]s/published/1992/pads/paper.ps

A Model for Collaborative Services in Distributed Learning.. - Hilt, Geyer (1997) (Correct) (5 citations)
protocol is required. We present an **ptimistic synchronization** scheme which provides consistency for by the model and not by a protocol. An **ptimistic synchronization** scheme assures consistency of the
www.informatik.uni-mannheim.de/informatik/pi4/publications/papers/hilt_idms97.ps.gz

Dynamic Load Balancing of a Multi-Cluster Simulator on.. - Schlagenhaft.. (1995) (Correct) (7 citations)
Time Warp [Jef85] is a well-known **optimistic synchronizati n** protocol for parallel simulations and with time over the CPUs. Due to the **optimistic synchronization** protocol which relies on a more or less
www.regent.e-technik.tu-muenchen.de/forschung/simul/englisch/../ps/pads95p.ps

Effect of Communication Overheads on Time Warp.. - Carothers, Fujimoto.. (1994) (Correct) (8 citations)
1 Introduction Time Warp [11] is an **optimistic synchronization** protocol that uses runtime detection of
www.cc.gatech.edu/grads/c/Chris.Carothers/./PAPERS/pads-94.ps

Efficient Object Sharing in Quantum-Based Real-Time Systems - Anderson, Jain, Jeffay (1998) (Correct) (3 citations)
CAS2 instruction succeeds. As with any **optimistic synchronization** scheme, concurrent operations may
www.cs.unc.edu/~anderson/papers/rtss98b.ps.Z

Interruptible Critical Sections - Johnson, Harathi (1994) (Correct) (6 citations)
interruptible critical sections (i.e.**optimistic synchronization**) as an alternative to purely blocking to purely blocking methods. Practical **optimistic synchronization** requires techniques for writing
ftp.cis.ufl.edu/pub/tech-reports/tr94/tr94-007.ps.Z

-body Algorithms for Interference Computation in Wireless.. - Felipe Perrone Dept (2000) (Correct) (1 citation)
[3, 18]While WiPPET relies on the **optimistic synchronization** protocol of Georgia Tech Time Warp
www.cs.dartmouth.edu/research/DaSSF/papers/MASCOTS00-nbody.pdf

Multiplexed State Saving For Bounded Rollback - Gomes, Unger, Cleary, Franks (1997) (Correct) (3 citations)
on the virtual time paradigm, is an **optimistic synchronization** algorithm in which LPs execute
www.informs-cs.org/wsc97papers/0460.PDF

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (RI) Google (Web) CSB DBLP

CiteSeer Find: thread synchronization and mutex Documents Citations

Searching for **thread synchronizati n and mutex**.
Restrict to: Header Title Order by: Citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CSB DBLP
22 documents found. **Order: citati ns weighted by y ar.**

SunOS Multi-thread Architecture - Powell, Kleiman, Barton, Shah.. (1991) (Correct) (60 citations)
threads. Shared memory S S Process 1 S **thread synchronization** variable Process 2 S Figure 1:
built using thread-local storage. **Thread synchronization** Threads synchronize with each other using
facilities include mutual exclusion (**mutex**) locks, condition variables and semaphores. For
www.ee.umd.edu/courses/enee647/threads/multi-thread.ps

Implementing Lightweight Threads - Stein, Shah (1992) (Correct) (40 citations)
section details how the library implements **thread synchronization**. The sixth section details how the
facilities include mutual exclusion (**mutex**) locks, condition variables, semaphores and
to. For example, in some cases acquiring an LWP **mutex** does not require kernel entry if there is no
uniser1.unl.csi.cuny.edu/~archives/postscripts/unix/impl_threads.ps

Managing Contention and Timing Constraints in a Real-Time.. - Matthew Lehr (1995) (Correct) (6 citations)
communication (RT-IPC) 5] and **thread synchronization** (RTSync) 10] facilities. RT-Mach
and priority inheritance. When a thread blocks on a **mutex** variable or when a message cannot be immediately
the Small Memory Manager is guarded by a realtime **mutex** variable to avoid the priority inversion problem
www.cs.virginia.edu/~vadb/publications/rtss95.ps

Developing A Real-Time Database: The Starbase Experience - Kim, Son (1997) (Correct) (1 citation)
communication (RT-IPC) 4] and **thread synchronization** (RT-Sync) 8] facilities. RT-Mach
and priority inheritance. When a thread blocks on a **mutex** variable or when a message cannot be immediately
the Small Memory Manager is guarded by a real-time **mutex** variable to avoid the priority inversion problem
cs.chungnam.ac.kr/~ykim/publications/chapter17.ps

Understanding Control Flow - With Concurrent Programming using.. - Buhr (1995) (Correct) (1 citation)
.268 12.2.1.2 **Thread Synchronization** and Mutual Exclusion .
.178 8.3.1 **Mutex** Calling **Mutex** .
with this implicit mutual-exclusion property as a **mutex** member (short for mutual exclusion member)and
plg.uwaterloo.ca/pub/uSystem/uC++book.ps.gz

Experimentation with Configurable, Lightweight Threads.. - Kaushik Ghosh.. (1993) (Correct) (1 citation)
offering constructs for thread fork, **thread synchronization**, shared memory between threads, etc.
the on-line configuration of the threads package's **mutex** locks 1 is shown to significantly improve the
about the operating system on 1 The terms `**mutex** lock' and `blocking lock' have been used
ftp.cc.gatech.edu/pub/coc/tech_reports/1993/GIT-CC-93-37.ps.Z

NICK BENTON, LUCA CARDELLI and C - Edric Fournet Microsoft (Correct)
calculus, messages, polyphonic C **synchronization, threads** Contents 1 Introduction 2 1.1
objectoriented form of threads and object-bound **mutex**es, but it has been provided at most as a veneer
includes a lock statement, which obtains the **mutex** associated with a given object during the
research.microsoft.com/~nick/polyphony/PolyphonyTOPLAS.A4.ps

IEEE November 10 - 13, 1999 San Juan, Puerto Rico - The Design And (Correct)
shift from sequential programming. **Thread synchronization** always causes problems. To address the
permits a user to create and join threads, and use **mutex** locks. With this kernel, students are able to
a layer of synchronization primitives that includes **mutex** locks, semaphores, mailboxes, reader-writer
fie.engrng.pitt.edu/fie99/papers/1032.pdf

Batons: A Sequential Synchronization Object - Tucker, Hart (Correct)
Object Multithreaded programming and **thread synchronization** are fundamental techniques in modern
with a set of three basic kernel objects: **mutex**es, events, and semaphores [2, 4]There are also
Pthreads standard [1]which uses just two objects: **mutex**es and condition variables. While the Win32 thread
www.halcyon.com/ast/dload/batons.pdf

A Simulator For A Multithreaded Processor - Adda Niar Bleuel (Correct)

Instruction set, pipelined execution, **thread synchronization** and creation. ABSTRACT This paper
(wait and syncp) that use a given resource as a **mutex**. The **mutex** is incremented by the wait and
syncp) that use a given resource as a **mutex**. The **mutex** is incremented by the wait and decremented by the
www.univ-valenciennes.fr/limav/niar/pub/rech/iasted.ps

Distribution as a Set of Cooperating Aspects - Position Paper Submitted (Correct)
`helpers' for a class: they take care of **thread synchr nization** over the methods of objects of that
declaration includes an number of selfex and **mutex** sets and a number of MethodManagers. Methods
however, will not deadlock. Methods included in a **mutex** set are mutually exclusive: if a method in a
web.iu-vannes.fr/~sadou/DOPP/fabry.ps

Pthreads and applications of mutex-abstraction - Hesselink, Jonker (2001) (Correct)
for the handling of threads and their **synchronization. Threads** with these primitives are called POSIX
Pthreads and applications of **mutex**-abstraction Wim H. Hesselink, Jan Eppo Jonker,
POSIX threads are light-weight processes with **mutex**es and condition variables for synchronization.
www.cs.rug.nl/~wim/pub/whh233.ps.gz

The Design and Construction of a User-Level Kernel for.. - Michael Bedy Steve (1999) (Correct)
shift from sequential programming. **Thread synchronization** always causes problems. To address the
permits a user to create and join threads, and use **mutex** locks. With this kernel, students are able to
a layer of synchronization primitives that includes **mutex** locks, semaphores, mailboxes, readerwriter locks,
www.cs.mtu.edu/~shene/edu/fie99-mtp.ps.gz

A False-Sharing Free Distributed Shared Memory Management Scheme - Alexander Chi Lai (2000) (Correct)
aggressive consistency, distributed **synchronization, thread**ed splay tree, false sharing 1.
locks datameaS0 seS0# tsdirexH0 insteH of an eS tra **mutex** (mutual exclusion) synchronization variable In
at a proceS/H whe that proceS7-locks a **mutex** that "guards"the data that is,the guarde data
search.ieice.org/2000/files/../pdf/e83-d_4_777.pdf

Synchronization Primitives for Threads - Hesselink, Jonker (2000) (Correct)
for the handling of threads and their **synchronization. Threads** with these primitives are called POSIX
support light-weight processes called threads, with **mutex**es and condition variables for synchronization.
is shown by means of invariants. Keywords thread, **mutex**, condition variable, signal, POSIX thread
www.cs.rug.nl/~wim/pub/whh217.ps.gz

Thread Synchronization - In The Last (Correct)
53 CHAPTER **Thread Synchronization** In the last chapter, we described
endl 14 ExitThread(0)15 }16 /got **Mutex**, begin critical section 17 cout Produce:
Producer-Consumer Problem. P C Producer Consumer **Mutex** Lock Shared Space 3.1 The Producer-Consumer
ftp.iftech.com/DevJournal/pdf/9904_pham_multithread.pdf

A User-level Checkpointing Library for POSIX Threads Programs - William Dieter James (Correct)
as if the checkpoint had not happened. **Thread synchronization** functions may not be safe to call in a
the signal handler. For example: 1. Thread 1 locks **mutex** M 2. Thread 1 blocks on a condition, unlocking
M 2. Thread 1 blocks on a condition, unlocking **mutex** M 3. Thread 2 locks **mutex** M 4. Both threads
www.dcs.uky.edu/~chkpt/pub/ftcs99.ps.gz

Simulating Fluids in Zero Gravity - Gabriel Somlo Computer (Correct)
work has been finished. As far as inter-**thread synchronization** is concerned, no thread will ever need to
available in a shared memory multiprocessor model: **mutex** locks, conditional locking, semaphores, etc. The
to solve this problem would be to assign a **mutex** lock to each node, forcing edge-processing
www.cs.colostate.edu/~somlo/publications/cisst98camera.ps.gz

Evaluation of a Real-Time eXtension(RTX) on Windows/NT - Yasu, Carcassi (Correct)
Events are inter-process and inter-**thread synchronization** objects that are used for signaling. A
shared memory, semaphores, event objects and **mutex** objects as inter-process communication. When
called RtCreateMutex creates a named or unnamed **mutex** object and returns its handle. A function called
atddoc.cern.ch/Atlas/Notes/../postscript/Note125.ps

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (RI) Google (Web) CSB DBLP